International Journal of Electrical and Computer Engineering (IJECE)
Vol. 10, No. 6, December 2020, pp. 6446~6452
ISSN: 2088-8708, DOI: 10.11591/ijece.v10i6.pp6446-6452 

# Improving the initial values of VFactor suitable for balanced modulus

**Kritsanapong Somsuk**
Department of Computer and Communication Engineering, Faculty of Technology,
Udon Thani Rajabhat University, Thailand

**Article Info**

***Article history:***

Received Mar 8, 2020
Revised Jun 4, 2020
Accepted Jun 20, 2020

***Keywords:***

Initial value
Loops
RSA
The last m digits
VFactor

**ABSTRACT**

The aim of this study is to estimate the new initial values of VFactor. In general, this algorithm is one of the members in a group of special proposed integer factorization algorithm. It has very high performance whenever the result of the difference between two prime factors of the modulus is a little, it is also called as balanced modulus. In fact, if this situation is occurred, RSA which is a type of public key cryptosystem will be broken easily. In addition, the main process of VFactor is to increase and decrease two odd integers in order to compute the multiplication until the targets are found. However, the initial values are far from the targets especially that the large value of the difference between two prime factors that is not suitable for VFactor. Therefore, the new initial values which are closer to the targets than the traditional values are proposed to decrease loops of the computation. In experimental results, it is shown that the loops can be decreased about 26% for the example of 256 bits-length of modulus that is from the small result of the difference between prime factors.



***Corresponding Author:***

Kritsanapong Somsuk,
Department of Computer and Communication Engineering,
Udon Thani Rajabhat University,
Udon Thani, 41000, Thailand.
Email: kritsanapong@udru.ac.th

## 1. INTRODUCTION

Integer factorization problem (IFP) has become one of the important issues since RSA [1] which is a type of asymmetric key crytosystem or public key cryptosystem [2] was proposed in 1978. The reason is that it can factor the modulus as prime numbers and then the private key kept secretly by owner is also recovered. That mean, this methodology is one of the tools for breaking RSA. At present, the length of the modulus should be assigned at least 1024 bits [3] to avoid attacking by intruders. However, if one of hidden parameters is weak, it is very easy to break this system, although bit-length of the modulus is very high. The examples of the weak parameters are consisting of a low private key [4-6], a high private key [7], a low prime factor [8, 9], all prime factors of p-1 or q-1 [10] which are small, where p and q are represented as prime factors of the modulus and the small result of p-q [11-14].

The aim of this paper is to modify one of the factoring algortihms which is suitable for the small result of p-q. The algorithm is called VFactor [15]. Both of two initial values are usually assigned for VFactor, x and y. Referring to the conditions, y is always decreased until y=q and x is always increased until x=p. However, the initial value of x and y are usually far from p and q, respectively. Therefore, in this paper, the new initial values of x and y which are very closer to the targets than the traditional values are proposed. The key concepts are from the considering the last m digits of p and q and the result of (p+q) mod 8 must be always equal to 0 when (n+1) mod 8 =0, where n is the modulus. Then, it implies that many unrelated values are removed from the computation.

The rest of the paper is organized as follows. In section 2, it mentions about the related works which consist of an overview of RSA, VFactor, tecniques to analyze the last m digits of p and q and conditions of p+q and p - q. In section 3, the proposed method which is the generating of the new initial values of x and y are presented. In section 4, the experimental results and discussion are mentioned. Finally, the conclusion will be discussed in the last session.

## 2. RELATED WORKS

### 2.1. RSA

RSA is a type of public key cryptography. It was proposed by three researchers, Ron Rivest, Adi Shamir, and Len Adleman in 1978. There are three main processes for this technique. The first process is a key generation and there are three steps to finish this process. Step 1 is to generate two prime numbers randomly, p and q, and then compute modulus, n=p*q, and euler totient value, $\Phi(n)=(p-1)*(q-1)$. The next step is to select a public key, t, with the following condition, $1<t<\Phi(n)$ and $\gcd(t, \Phi(n))=1$. After that, a private key, h, can be computed from $t*h \bmod \Phi(n)=1$ by using some of extended euclidean algorithms [16-19]. The second process is an encryption process to convert original plaintext, m, as ciphertext, c, from the equation: $c=m^t \bmod n$. The last process is a decryption process to recover m by using the equation: $m=c^h \bmod n$. Generally, it is very difficult to break this system when bit-length of n is at least 1024 and all parameters are strong. In contrast, RSA becomes easily attacked when some of parameters are weak. One of the weak parameters is the small value of p-q. There are various tecniques which are suitable for this condition. One of them is VFactor which is a type of integer factorization algorithm.

### 2.2. VFactor and improvement

VFactor is one of integer factorization algorithms. This algorithm which was proposed by Sharma et al., has very high performance when the result of p-q is very close to 0. Two odd integers are chosen as the initial values. The first value is $y=\lfloor\sqrt{n}\rfloor$ but y may be decreased by 1 to ensure that it is an odd number when it is an even number. The other value is x=y+2. The main process is to compute m=x*y. In fact, if m=n, then it implies that x and y are two large prime factors of n. However, it is divided into two cases. The first case is m > n while y is too large, then y has to be decreased by 2. On the other hand, the second case is occurred when m<n, x is too small and it must be increased by 2. In fact, the process is continuously repeated until m=n is found. Moreover, the modified algorithms of VFactor were proposed to remove some loops and time. MVFactor [20] is the technique to decrease both of x and y out of the computation when the last digit is equal to 5. In fact, the odd integers which the last digit is equal to 5, except 5, are not certainly a prime number, because 5 divides all of them. Later, MVFactorV2 [21] was proposed. The key is to choose only x and y which must be written in the following form: 6k+1 or 6k-1, where $k \in Z$. Moreover, the last digit of them must not be equal to 5. Therefore, the odd integers which the last digit is 5 and can not be written as the form 6k+1 or 6k-1 are certainly not a prime number. Table 1 is shown the steps of increasing the odd integer to skip unrelated values. Furthormore, the information in the table is also selected to consider the decreasing steps.

Table 1. The increasing steps of the odd integer that may be a prime number

| Row | LSG(n) | n mod 6 | Increasing Steps |
|---|---|---|---|
| 1 | 1 | 5 | 0 |
| 2 | 3 | 1 | 2 |
| 3 | 5 | 3 | None |
| 4 | 7 | 5 | 4 |
| 5 | 9 | 1 | 2 |
| 6 | 1 | 3 | None |
| 7 | 3 | 5 | 4 |
| 8 | 5 | 1 | None |
| 9 | 7 | 3 | None |
| 10 | 9 | 5 | 6 |
| 11 | 1 | 1 | 2 |
| 12 | 3 | 3 | None |
| 13 | 5 | 5 | None |
| 14 | 7 | 1 | 6 |
| 15 | 9 | 3 | None |
| 16 | 1 | 5 | 4 |

The information in Table 1 shows the increasing steps of the odd integer that may be a prime number. All prime numbers, except 2 and 3, must be usually rewritten as two forms consisting of 6k-1 and 6k+1. That mean the integer which its condition is equal to the data in row 3rd, 6th, 8th, 9th, 12th, 13th and 15th of this table is not certainly a prime. The reason is that the form of some of them is 6k+3 or the last digit is 5. Therefore, if x has to be increased or y has to be decreased, then the increasing steps in this table can be chosen to remove the odd integers which are not certainly a prime number. For example, assume that the lastest value of x has the last 2 digits as 63 and the condition is in 7th row, the next value should have the last 2 digits as 69.

### 2.3. Analyzing the last m digits of p and q

In 2017, [22] the technique to analyze all last m digits of p and q was proposed. After finding all of them, many unrelated integers are removed out of the computation. In fact, they are chosen to leave some loops of FFA. Assuming some values which may be the last m digits of p and q are disclosed. There are two rules for analyzing the others which may be also the last m digits of p and q as follows: (Assigning $p_m$ is represented as the last m digits of p and $q_m$ is represented as the last m digits of q.

Rule 1: If the last digit of p and q are same, the other pairs can be computed from $p_{m'}=(p_m+10^{m-1}) \bmod 10^m$ and $q_{m'}=(q_m+9*10^{m-1}) \bmod 10^m$

*Example 1* Assuming $p_2=11$, $q_2=31$ (the last 2 digits of 11*31=41), then $p_{2'}=21$ and $q_{2'}=21$ (the last 2 digits of 21*21=41)

Rule 2: If the last digit of p and q are different, the other pairs can be computed from $p_{m'}=(p_m+k_1*10^{m-1}) \bmod 10^m$ and $q_{m'}=(q_m+k_2*10^{m-1}) \bmod 10^m$, where $((p_m \bmod 10)*k_2+(q_m \bmod 10)*k_1) \bmod 10=0$

In fact, after finding all last m digits of p and q, all possible results of the last m digit of p+q and p-q are also disclosed. Assuming U is represented as the set of all possible values of the last m digits of p+q and V is represented as the set of all possible values of the last m digits of p-q. Example 2 is shown the way to find all members for both of them.

*Example 2* Finding U and V for all values of n that the last 2 digits are 83

*Sol.* In general, both of rule 1 and rule 2 are the key to find all members of U and V. In fact, all pairs of the last 2 digits of p and q are as follows: (11, 53), (21, 23), (31, 93), (41, 63), (51, 33), (61, 3), (71, 73), (81, 43), (91, 13), (1, 83), (17, 99), (27, 29), (37, 59), (47, 89), (57, 19), (67, 49), (77, 79), (87, 9), (97, 39) and (7, 69). Therefore, U and V are as follows:

U={04, 16, 24, 36, 44, 56, 64, 76, 84, 96}
V={02, 18, 22, 38, 42, 58, 62, 78, 82, 98}

In addition, after U and V are found, the initial value of u, $u_i$, which is begun as $2\lceil\sqrt{n}\rceil$ can be reestimated. The last m digits of $u_i$ should be one of the members in U. That means it can be increased whenever the result is still not a member of U.

### 2.4. Analyzing the initial value of p-q

The initial value of p-q should be usually begun as 0, p=q. However, real value of p-q is very far from the initial value. In 2018, [23] the equation to estimate the new initial value of v, $v_i$, was proposed. In fact, before using the equation, all last m digits of p and q must be disclosed. In addition, $v_i$ can be computed from the following equation: $v_i=\lceil\sqrt[4]{d^2*n}\rceil$, where d is the distance between the traditional value of $u_i$ and the new value of $u_i$. In addition, the new values of $u_i$ and $v_i$ can be also selected to decrease time for some other factorization algortihms. For example, in 2019, this technique is chosen to combine with trial divition algorithm (TDA) [24]. Before applying this method with TDA, the first divisor is usually begun as $\lfloor\sqrt{n}\rfloor$. On the other hand, it may be assigned as the integer which is less than this value when it is applied with TDA.

### 2.5. Analyzing the remainder of (p+q) mod 8

In [25], it is found that if the result of (n+1) mod 8=0, then the result of (p+q) mod 8 must be always equal to 0. Therefore, only pattern of p+q which is in the condition will be included to remove some loops of the computation.

## 3. THE PROPOSED METHOD

In this section, the new initial values to both of x and y for Vfactor are proposed to decrease the certainly unrelated values out of the computation. In general, the traditional initial value of x is

the minimum odd integer which is larger than $\sqrt{n}$. On the other hand, after the last m digits of n are analyzed, it can be estimated as $q_i$ when $q_i$ is an odd number or $q_i$+1 when $q_i$ is an even number. In addition, the traditional initial value of y is the maximum odd integer which is still less than $\sqrt{n}$. The same reason with above condition, the new value can be estimated as the maximum odd integer which is less than $\frac{n}{q_i}$. Furthermore, if the concepts of MVFactor and MVFactorV2 are also included, then the last digit must not be equal to 5 and the forms of them must be always 6k+1 or 6k-1.

Algorithm 1: The new initial values of x and y

```
Input: n, ui, vi
1. qi ← (ui + vi)/2
2. IF qi%2 == 0 Then
3.    qi ← qi+1
4. End IF
5. x ← qi
6. x10 ← x % 10
7. x6 ← x % 6
8. IF (x10 == 5 and x6 == 3) OR (x10 == 1 AND x6 == 3) OR (x10 == 7 AND x6 == 3) OR (x10 ==5
   AND x6 ==5) OR (x10 == 9 AND x6 == 3) Then
9.     x ← x+2
10.Else IF (x10 == 5 AND x6 == 1) OR (x10 == 3 AND x6 == 3) Then
11.     x ← x+4
12.End IF
13. y ← ⌊n/x⌋
14. IF y%2 == 0 Then
15.    y ← y - 1
16. End IF

17. y10 ← y %10
18. y6 ← y % 6
19. IF (y10 == 5 and y6 == 3) OR (y10 == 1 AND y6 == 3) OR (y10 == 5 AND y6 == 1) OR (y10 ==3
    AND y6 ==3) OR (y10 == 9 AND y6 == 3) Then
20.     y ← y - 2
21. Else IF (y10 == 7 AND y6 == 3) OR (y10 == 5 AND y6 == 5) Then
22.     y ← y-4
23. End IF
Output: The new initial values of x and y
```

Algorithm 2: The new initial values of $u_i$ and $v_i$

```
Input: n (n+1 mod 8=0), U, V
1. ui ← 2⌈√n⌉
2. t ← ui
3. IF the last m digits of ui is not a member of U Then
4.     Increaning ui until the last two digits are equal to one of the members in U
5. End IF
6. While ui mod 8 is not equal to 0 do
7.      Replacing the last m digits of ui by the next member of U
8.  End While
9. d ← ui - t
10. vi ← ⌈∜(d² * n)⌉
11. IF the last m digits of vi is not a member of V Then
12.     Increaning vi until the last two digits are equal to one of the members in V
13. End IF
Output: The new initial values of ui and vi
```

*Example 3:* Finding the new initial values of x and y when n=2620361083 (56533*46351) by considering the last 2 digits of n=83 and using Algorithm 1

*Sol.* Before using Algorithm1, $u_i$ and $v_i$ must be computed. Usually $u_i$=2 $\lceil\sqrt{2620361083}\rceil$=102380. However, the last 2 digits is 80 which is not a member of U. Therefore, $u_i$ can be increased as 102384, and then d=4. In addition, $v_i$=2 $\lceil\sqrt[4]{4^2 * 2620361083}\rceil$=906. Nevertheless, the last 2 digits is 06 which is not a member of V. Then, $v_i$ can be increased as 918. Therefore, each step in Algorithm 1 is as follows:

Step 1: $q_i=\frac{102384+918}{2}$ =51651
Step 2-4: $q_i$ is not changed, because $q_i$ % 2=1
Step 5: x=51651
Step 6-7: $x_{10}$=1 and $x_6$=3
Step 8-12: x=51653
Step 13: y= $\left\lfloor\frac{2620361083}{51653}\right\rfloor$ =50730
Step 14-16: y is changed as 50729, because y % 2=0
Step 17-18: $y_{10}$=9 and $y_6$=5
Step 19-23: y is not changed, because both of $y_{10}$ and $y_6$ are not matched with the conditions.

Therefore, the new initial values are x=51651 and y=50729

In fact, the traditional initial values in example 3 are as follows: $y=\lfloor\sqrt{2620361083}\rfloor=51189$ and x=y+2=51191. Then, x' s loops are decreased as $\frac{51653-51191}{2}=231$ and y' s loops are decreased as $\frac{51189-50729}{2}=230$. Therefore, total loops are left out the computation about 461 whenever the new initial values of x and y are chosen instead of the tradition values.

Furthermore, total loops are more decreased when m is large. The reason is that the characteristic of n is analyzed more deeply. In contrast, loops are not reduced when the last m digits of $2\lceil\sqrt{n}\rceil$ is the member of U, because d is equal to 0. Therefore, both of $u_i$ and $v_i$ are not changed.

Moreover, the idea in [25] can be selected to apply with the proposed method when the result of (n+1) mod 8 is equal to 0. In fact, d is expanded, because $u_i$ can be modified in the conditions of U and u mod 8=0. However, both of $u_i$ and $v_i$ must be improved before using Algorithm 1. For Algortithm 2, it shows the process to improve $u_i$ and $v_i$ when (n+1) mod 8 is equal to 0.

*Example 4:* Finding the new initial value of x and y when n=3801472783 (63073*60271) by considering the last 2 digits of n=83 and using Algorithm 1 and Algorithm 2

*Sol.* First, the result of (n+1) mod 8 have to be determinded. Because the result is 0, then the pattern of (p+q) mod 8 must be also 0.

The process to find the new values of $u_i$ and $v_i$ by using Algorithm 2 is as follows: Usually $u_i=2\left\lceil\sqrt{3801472783}\right\rceil$ = 123314. However, the last 2 digits is 14 which is not a member of U. Therefore, $u_i$ can be increased as 123316. In contrast, 123316 mod 8=4 $\neq$ 0, then next value of $u_i$ should be 123324. However, 123324 mod 8=4 $\neq$ 0, then next value of $u_i$ should be assigned as 123336. Because 123336 mod 8=0, it is the new value of $u_i$, d=22. In addition, $v_i=2\left\lfloor\sqrt[4]{22^2*3801472783}\right\rfloor=2330$. However, the last 2 digits is 30 which is not a member of V. Therefore, $v_i$ can be increased as 2338. Therefore, each step in Algorithm 1 is as following:

Step 1: $q_i=\frac{123336+2338}{2}=62837$
Step 2-4: $q_i$ is not changed, because $q_i$ % 2=1
Step 5: x=62837
Step 6-7: $x_{10}$=7 and $x_6$=5
Step 8-12: x is not changed, because both of $x_{10}$ and $x_6$ are not matched with the conditions.
Step 13: y= $\left\lfloor\frac{3801472783}{62837}\right\rfloor$ = 60497
Step 14-16: y is not changed, because y % 2=1
Step 17-18: $y_{10}$=7 and $y_6$=5
Step 19-22: y is not changed, because both of $y_{10}$ and $y_6$ are not matched with the conditions.

Therefore, the new initial values are x=62837 and y=60497.

The traditional initial values in example 4 are usually as following: y=$\lfloor\sqrt{3801472783}\rfloor$=61656=61655 (y is always an odd number) and x=y+2=61657. Then, x' s loops are decreased as $\frac{62837-61655}{2}=590$ and y' s loops are decreased as $\frac{61655-60497}{2}=579$. Therefore, total loops are left out the computation about 1169 whenever the new initial values of x and y are chosen instead of the tradition values. In fact, in this example, assuming the concept in [25] is not chosen to combine with the proposed method, $u_i$=123316, d=2 and $v_i$=718. Hence, the total loops are decreased only 359. Therefore, loops are more decresed about three times. However, this technique can not be appied with n in example 3, because n+1 mod 8=4 $\neq$ 0.

## 4. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, experimental results and discussion will be presented. In fact, the last 2 digits of all values of n in the experiment are 83, because U and V are already considered in related works section to skip this process. However, if the other cases of n are occurred, both of U and V must be considered at first. Furthermore, n+1 mod 8=0 for all values of n in this section is selected to include the idea behind Algorithm 2. The experiment is about the comparison of decreasing loops between using only Algorithm 1 and the combination between Algorithm 1 with Algorithm 2. In addition, bit-length of n which is randomly chosen in this experiment consist of 32, 64, 128, 256, 512 and 1024. Moreover, 50 values of n are chosen for the same bit-length to find the average. However, the condition of n in this session is that d must not be equal to 0. The reason is that the new initial values for both of Algorithm 1 and the combination between Algorithm 1 and Algorithm 2 are still equal to the tradition initial values.

The information in Table 2 shows that if (n+1) mod 8=0 and $d \neq 0$, decreasing loops of the computation by using the combination between Algorithm 1 and Algorithm 2 are much higher than using Algorithm 1 only. In addition, it is larger than the other about two times. Therefore, to ensure that all hidden parameters are strong, the result of (n+1) mod 8 should not be equal to 0. In fact, the probability is equal to 0.25 that the result of (n+1) mod 8=0, n is selected randomly.

Table 2. The comparion of decreasing loops between two proposed techniques for (n+1) mod 8=0

| Bit-length of n | Decreasing Loops | |
|---|---|---|
| | Algorithm 1 | Algorithm 1+Algorithm 2 |
| 32 | 287 | 752 |
| 64 | 77683 | 257612 |
| 128 | 9656643962 | 18065926601 |
| 256 | $4.31*10^{19}$ | $8.07*10^{19}$ |
| 512 | $7.23*10^{38}$ | $1.35*10^{39}$ |
| 1024 | $3.06*10^{77}$ | $5.72*10^{77}$ |

However, if n is larger than 1024 bits and all hidden parameters are strong, VFactor and all improving algorthms, including the proposed methods and the result of (n+1) mod 8=0, do not still break RSA within a polynomial time. The example is shown as follows: Assuming n=2930609108682909796272667852321420978572050300727269278625554157598777850283195 60086299868745423605959016054076895558512635625836613350661870819438814212383 (256 bits-length), after estimating the new initial values, the decreasing loops are about $7.98 *10^{19}$. However, the total loops are $2.34*10^{37}$. Therefore, after using the proposed method, loops are decreased only $3.14*10^{-11}$ % that is very too small. In contrast, the proposed method become high performance when p is close to q. The example is shown as follows: Assuming n=1944566304086206135271835788021169282895 194456630408620613127183578802116928247=37813381109874874999245217886281867608252777770 85543433317873242209185747938 3 (256 bits-length), after estimating the new initial values, the decreasing loops are about $2.78 *10^{19}$. However, the total loops are $1.06*10^{20}$. Therefore, after using the proposed method, loops are decreased about 26% that are very high. Therefore, the ratio of the decreasing loops is based on the characteristics of p and q and the proposed method is suitable for a small result of p and q.

## 5. CONCLUSION

The new initial values for VFactor are assigned to leave the unrelated values out of the computation. The key is to choose the concept of the consideration of the last m digits of p and q. In fact, after all of them are found, the patterns of the last m digits of p+q and p-q are also disclosed. Both of them are the keys to estimate the new initial values for this method. Moreover, this technique is also included with the other pattern of p+q that the result of (p+q) mod 8 is always equal to 0 when then result of (n+1) mod 8 is 0. Two algorithms are proposed in this paper. The first is called Algorithm 1 which can be applied with all values of n. However, before using this algorithm, U and V must be calculated. Another one is called Algorithm 2. This algorithm is chosen to support Algorithm 1 when the result of (n+1) mod 8=0. The experimental results show that if (n+1) mod 8=0, the decreasing loops of the computation by using the combination between Algorithm 1 and Algorithm 2 are higher than using only Algorithm 1 about two times. Furthormore, in experimental results, it is shown that the loops can be decreased 26% in the example of 256 bits-length of n when the difference between prime factors is small.

## REFERENCES


[1] R. L. Rivest, et al., "A method for obtaining digital signatures and public key cryptosystems," *Communications of ACM*, vol. 21, pp. 120-126, 1978.

[2] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644-654, 1976.

[3] Y. K. Kumar and R. M. Shafi, "An efficient and secure data storage in cloud computing using modified RSA public key cryptosystem," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 1, pp. 530-537, 2020.

[4] M. J. Wiener, "Cryptanalysis of short RSA secret exponents," *IEEE Transactions on Information Theory*, vol. 36, no. 6, pp. 553-558, 1990.

[5] D. Boneh and G. Durfee, "Cryptanalysis of RSA with Private Key d less than $N^{0.292}$," *International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 1-11, 1999.

[6] M. E. Wu, et al., "On the Improvement of Wiener Attack on RSA with Small Private Exponent," *The Scientific World Journal*, vol. 2014, pp. 1-9, 2014.

[7] K. Somsuk, "The New Equation for RSA's Decryption Process Appropriate with High Private Key Exponent," *Proceedings of International Computer Science and Engineering Conference*, pp. 1-5, 2017.

[8] M. Sahin, "Generalized Trial Division," *International Journal of Contemporary Mathematical Science*, vol. 6, no. 2, pp. 59-64, 2011.

[9] N. Lal, et al., "Modified trial division algorithm using KNJ-factorization method to factorize RSA public key encryption," *Proceedings of IEEE International Conference on Contemporary Computing and Informatics*, pp. 992-995, 2014.

[10] S. Sarnaik, et al., "Comparative study on Integer Factorization Algorithm-Pollard's RHO and Pollard's P-1," *Proceedings of the International Conference on Computing for Sustainable Global Development*, pp. 677-679, 2015.

[11] M. E. Wu, et al., "On the improvement of Fermat factorization using a continued fraction technique," *Future Generation Computer Systems*, vol. 30, no. 1, pp. 162-168, 2014.

[12] K. Omar and L. Szalay, "Sufficient conditions for factoring a class of large integers," *Journal of Interdisciplinary Mathematical*, vol. 13, no. 1, pp. 95-103, 2010.

[13] S. Vynnychuk, et al., "Application of the basic module's foundation for factorization of big numbers by the fermat method," *Eastern European Journal of Enterprise Technologies*, vol. 6, pp. 14-23, 2018.

[14] J. Mckee, "Speeding Fermat's factoring method," *Mathematics of Computation*, vol. 68, no. 228, pp. 1729-1737, 1999.

[15] P. Sharma, et al., "Notice of Violation of IEEE Publication Principles: Modified Integer Factorization Algorithm Using V-Factor Method," *Proceedings of International Conference on Advanced Computing & Communication Technologies*, pp. 423-425, 2012.

[16] D. Phiamphu and P. Saha, "Redesigned the Architecture of Extended-Euclidean Algorithm for Modular Multiplicative Inverse and Jacobi Symbol," *Proceedings of International Conference on Trends in Electronics and Informatics*, pp. 1345-1349, 2018.

[17] J. Zhou, et al., "Extended Euclid algorithm and its application in RSA," *Proceedings of International Conference on Information Science and Engineering*, pp. 2079-2081, 2010.

[18] M. M. Asad, et al., "Performance Analysis of 128-bit Modular Inverse Based Extended Euclidean Using Altera FPGA Kit," *Procedia Computer Science*, vol. 160, pp. 543-548, 2019.

[19] Q. Zhou, et al., "How to securely outsource the extended euclidean algorithm for large-scale polynomials over finite fields," *Information Sciences*, vol. 512, pp. 641-660, 2020.

[20] K. Somsuk and S. Kasemvilas, "MVFactor: A method to decrease processing time for factorization algorithm," *Proceedings of International Computer Science and Engineering Conference*, pp. 339-342, 2013.

[21] K. Somsuk, "MVFactorV2: An improved integer factorization algorithm to speed up computation time," *Proceedings of International Computer Science and Engineering Conference*, pp. 308-311, 2014.

[22] K. Somsuk and K. Tientanopajai, "An Improvement of Fermat's Factorization by Considering the Last m Digits of Modulus to Decrease Computation Time," *International Journal of Network Security*, vol. 19, no. 1, pp. 99-111, 2017.

[23] K. Somsuk, "The improvement of initial value closer to the target for Fermat's factorization algorithm," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 21, no. 7-8, pp. 1573-1580, 2018.

[24] K. Somsuk, et al., "Estimating the new Initial Value of Trial Division Algorithm for Balanced Modulus to Decrease Computation Loops," *Proceedings of International Joint Conference on Computer Science and Software Engineering*, pp. 143-147, 2019.

[25] Y. B. Hammad, et al., "RAK factoring algorithm," *Australasian Journal of Combinatorics*, vol. 33, no. 1, pp. 291-305, 2005.